Entry Exam – Hogwarts

[1. Counting Dementor 1](#_Toc181368958)

[2. Order of the Phoenix Votes 2](#_Toc181368959)

[3. Hogwarts Code Cipher 2](#_Toc181368960)

[4. Hogwarts Library Archives 3](#_Toc181368961)

[5. Epic Wizarding Duels 3](#_Toc181368962)

[6. Auror Battle Formation 4](#_Toc181368963)

*May your problem-solving skills be as sharp as the Elder Wand and may the wisdom of Dumbledore guide your code!*

# Counting Dementor

Harry is training his Patronus Charm spell by fighting Dementors in Azkaban. Given that he fights n Dementors in the first hour and increases his count by m every subsequent hour, determine the total number of Dementors he will have fought after h hours.

**Input**: Three integers:

* + **n** represents the initial count
  + **m** represents the increase of the count.
  + **h** represents the number of hours.

**Constraints**: n increases by m every hour.

|  |  |
| --- | --- |
| Input | Output |
| 5, 2, 3 | 21 |
| 7, 1, 2 | 15 |
| 10, -1, 1 | 10 |
| 8, -2, 3 | 18 |

*...* *gliding, wraithlike Dark creature, widely considered to be one of the foulest…*

# Order of the Phoenix Votes

During the meeting of the Order of the Phoenix, members are voting on whether to proceed with a dangerous mission against Voldemort. Each member can vote "Yes," "No," or "Abstain." Given an array of votes, determine the decision of the Order – Yes, No, Abstain, or Tie.

* Input: An array of strings (votes).

|  |  |
| --- | --- |
| Input | Output |
| ["Yes", "No", "Yes", "Abstain", "Yes"] | Yes |
| ["No", "No", "Yes"] | No |
| ["Yes", "No"] | Tie |
| ["Abstain"] | Abstain |
| ["No", "No", "Abstain", "Abstain", "Yes", "Yes"] | Tie |
| ["No", "Abstain", "Abstain"] | No |

*"No, Harry, the meeting's only for members of the Order." ― Molly Weasley*

# Hogwarts Code Cipher

The wizards at Hogwarts have a secret code used to send encrypted messages across the magical world. The code replaces each letter in a message with the letter that is n places before it in the alphabet. Write a program that deciphers the message.

* Input: A string **s** (1 ≤ |s| ≤ 100) representing the encrypted message and an integer **n** (1 ≤ n ≤ 25) representing the shift.

**Input/Output**:

|  |  |
| --- | --- |
| Input | Output |
| 'Uifsf!jt!b!tfdsfu"', 1 | There is a secret! |
| 'vvhhdkdvkkdvvhh$',3 | sseeahashhassee! |
| 'Ibssz!Qpuufs', 1 | Harry Potter |

*"We are only as strong as we are united, as weak as we are divided." — Albus Dumbledore*

# Hogwarts Library Archives

The Hogwarts Library contains a vast array of magical records. Write a program that finds the first and last occurrence of a specific record in the archives. If the record is missing, print "Record not found."

* Input: An array of strings representing the records and a string representing the search term.

**Input/Output**:

|  |  |
| --- | --- |
| Input | Output |
| ["Harry", "Hermione", "Harry", "Ron"], "Harry" | First Occurrence: 0  Last Occurrence: 2 |
| ["Ginny", "Luna", "Neville", "Ginny"], "Ginny" | First Occurrence: 0  Last Occurrence: 3 |
| ["Ron", "Hermione", "Luna"], "Dumbledore" | Record not found |
| ["Dobby", "Ron", "Hermione", "Harry"], "Harry" | First Occurrence: 3  Last Occurrence: 3 |

*"Dobby is always wanting to help Harry Potter." — Dobby*

# Epic Wizarding Duels

During a wizarding duel, every spell has a counterspell:

* If an opponent casts a jinx represented by /, the hero counters with a defensive charm \.
* If an opponent casts a powerful curse represented by !, the hero counters with another powerful curse !.
* If an opponent uses ancient magic represented by <, the hero counters with protective magic >.

If a sequence of spells and counterspells is balanced in the correct order, the duel is considered epic. Determine if a given sequence is epic duel.

* Input: A string sequence of spells and counterspells.

**Input/Output**:

|  |  |
| --- | --- |
| Input | Output |
| "/\/\//\/\\" | Epic |
| "//!!\//<!!>\\\" | Epic |
| "///\\" | Not Epic |
| "<!>!" | Not Epic |
| "/</!!>\\" | Not Epic |

*"Wands at the ready… let the duel begin!" — Gilderoy Lockhart*

# Auror Battle Formation

The Auror team is arranged in a specific formation for their tactical strikes against dark forces. They maintain this formation throughout the operation, executing various commands as they engage in battle. Write a program that handles the following commands and returns the new formation after each valid command.

Input: An initial array of integers representing Auror IDs and a list of commands.

**Commands**:

* **"destroy [index]":** The Auror at the given index is defeated and removed from the formation.
* **"swap [index1] [index2]":** Swap the positions of two Aurors with the given indices.
* **"add [ID]":** A new Auror joins the formation, positioning at the end.
* **"insert [ID] [index]":** Insert a new Auror with the given ID at a specific position.
* **"center":** Display the Auror(s) in the center of the formation. If there is an even number of Aurors, display the middle two.

Note: Ensure the commands handle edge cases, e.g., destroying a non-existent rider or inserting at a position out of bounds should take no action.

|  |  |
| --- | --- |
| Input | Output |
| [1, 2, 3, 4, 5],  ["destroy 3",  "swap 0 1",  "add 6",  "center"] | 1 2 3 5  2 1 3 5  2 1 3 5 6  3 |
| [1, 2, 3, 4, 5],  ["add 6",  "swap 0 5",  "swap 1 4",  "swap 2 3",  "swap 2 100",  "swap 2 2",  "center"] | 1 2 3 4 5 6  6 2 3 4 5 1  6 5 3 4 2 1  6 5 4 3 2 1  4 3 |
| [1, 2],  ["insert 3 2",  "center",  "destroy 1",  "destroy 2",  "center"] | 1 2 3  2  1 3  1 3 |

*"Hogwarts is threatened! Man the boundaries, protect us, do your duty to our school!"  
— Professor McGonagall*